A Lightweight Distributed Solution to Content Replication in Mobile Networks

C.-A. La, P. Michiardi
EURECOM
Sophia Antipolis, France
firstname.lastname@eurecom.fr

C. Casetti, C.-F. Chiasserini, M. Fiore
Politecnico di Torino
Torino, Italy
firstname.lastname@polito.it

ABSTRACT

Performance and reliability of content access in mobile networks is conditioned by the number and location of content replicas deployed at the network nodes. Facility location theory has been the traditional, *centralized* approach to study content replication: computing the number and placement of replicas in a network can be cast as an uncapacitated facility location problem.

The endeavour of this work is to design a *distributed, lightweight* solution to the above joint optimization problem, while taking into account the network dynamics. In particular, we devise a mechanism that lets nodes share the burden of storing and providing content, so as to achieve load balancing, and decide whether to replicate or drop the information so as to adapt to a dynamic content demand and time-varying topology. We evaluate our mechanism through simulation, by exploring a wide range of settings and studying realistic content access mechanisms that go beyond the traditional assumption matching demand points to their closest content replica. Results show that our mechanism, which uses *local measurements only*, is: (i) extremely precise in approximating an optimal solution to content placement and replication; (ii) robust against network mobility; (iii) flexible in accommodating various content access patterns, including variation in time and space of the content demand.

1. INTRODUCTION

Research activity in the networking field is pursuing the idea that networks should provide access to contents, rather than to hosts. This idea is manifested in content distribution networks based on either peer-to-peer networks, or on an infrastructure of large storage nodes located close to edge networks. In this paper, we explore this concept with respect to wireless networks where nodes can exploit device-to-device communications. We highlight that content is being stored in nodes that move and that the content itself moves and is replicated in anticipation of being accessed. The purpose is to re-assess content distribution with respect to wireless networks in which content demand and topology are dynamically changing.

Previous research has focused on techniques that enhance performance and reliability of content access in wireless systems. Content caching and replication have been shown to be effective in achieving these goals (see, e.g., [1] for a survey on the topic). Every mobile device can potentially participate to content caching or replication by storing data which can be made available to other users through device-to-device communication over one of its wireless interfaces, e.g., IEEE 802.11 or Bluetooth.

Due to the storage capacity constraints of cache servers, effective cache eviction policies have traditionally been the subject of a large body of research in computer science in general, and in the context of wireless networks in particular [2–4]. Decision problems concerning the location of content replicas and, optionally, the number of replicas to deploy in a network have also generated a large number of works: prominent examples of such studies are available for wireline networks (e.g., content distribution networks) [5] and, to a lesser extent, for wireless networks [6].

In this work we focus on *content replication* in the context of mobile wireless networks in which users create a cooperative environment. The very nature of wireless content access and node mobility introduces several problems to content replication. *Optimal replica placement* is one of those: selecting the location that is better suited to store content is difficult, especially when the network is dynamic. However, optimal replica placement is just a facet of content distribution: another prominent issue is *how many content replicas* should be made available to mobile nodes. Clearly, network and content demand dynamics affect the solution of these two aspects. Furthermore, decisions on the placement and number of replicas to be deployed in a network are tightly related problems: intuitively, the latter introduces a feedback loop to the former as every content replication triggers a new instance of the placement problem.

Traditionally, the above content replication problems have been studied through the lenses of classic Facility Location Theory [7]: optimal placement can be cast as the *uncapacitated k-median* problem, whereas the joint optimization of placement and number of replicas can be studied as an *uncapacitated facility location* problem. Both these problems are NP-hard for general network topologies; furthermore, the above formulations do not tackle the problem of *how users can access contents.*

In our previous work [8], we show preliminary results indicating that a uniformly distributed replica placement can be well approximated using distributed store-and-forward mechanisms, in which nodes store content only temporarily. The endeavor of this work is to extend our previous study and target the *joint problem* (i) of establishing the number of replicas to deploy in a dynamic network, (ii) of finding their most suitable location, and (iii) of letting users efficiently access stored content. In particular, we address (i) and (ii) so as to achieve load balancing, that is, to let the network nodes evenly share the burden of storing and providing content.

Instead of designing approximation algorithms of the optimal solution to facility location problems which require global (or extended) knowledge of the network [5, 9], we integrate our store-and-forward mechanism with a distributed replication algorithm that bases its decisions on local measurements only and aims at evenly distributing among nodes the demanding task of being a replica provider. Also, we consider different content query/reply mechanisms and evaluate their performance in conjunction with the schemes used for content replication and placement.

As a result, we show that both optimal placement and content replication can be approximated through a lightweight, distributed scheme which adapts to different initial distributions of replicas and to variation in time and in space of content demand, while being robust against network dynamics.

2. BACKGROUND AND PROBLEM STATEMENT

As remarked above, the problem of content replication and caching has received a lot of attention in the past due to its importance in enhancing performance, availability and reliability of content access for Web-based applications. Here, we inherit the problem of replication typical of the wired-Internet and we discuss why the dynamic nature of wireless networks introduces new challenges with respect to the wireline counterpart. Note that, although several cache replacement policies have been proposed in the context of mobile ad-hoc networks [2–4], in this paper we focus on *replication* and *replica placement* problems, i.e., we view content replication as a process of its own, rather than a by-product of a query/caching mechanism [1].

Let us now define the context of our work. We investigate a scenario involving users equipped with devices offering Internet broadband connectivity as well as device-to-device communication capabilities (e.g., through IEEE 802.11). Although we do not concern ourselves with the provision of Internet access in ad hoc wireless networks, we remark that broadband connectivity is where new content is fetched from (and updated).

In order to provide a basic description of the system, we focus on content being represented by a *single* information object. The mechanisms we describe can then be extended to multiple objects. We assume the object to be tagged with a validity time, and originally hosted on a server in the Internet, which can only be accessed through the broadband access we hinted at. We then consider a *cooperative network environment* composed of a set $V = \{v(1), ..., v(N)\}$ of mobile nodes. A node $v(j)$ wishing to access the content first tries to retrieve it from other devices; if its search fails, the node downloads a fresh content replica from the Internet server and temporarily stores it for a period of time $\tau_{v(j)}$, termed *storage time*. For simplicity of presentation, in the following we assume $\tau_{v(j)} = \tau, \forall j \in V$. During the storage period, $v(j)$ serves the content to nodes issuing requests for it and, possibly, downloads from the Internet server a fresh copy of the content if its validity time has expired. We assume that a node $v(i)$, which at a given time t does not store any copy of the content and which will later be referred to as "content consumer", issues queries at a rate $\lambda_{v(i)}(t)$.

To achieve load balancing, at the end of the storage time $v(j)$ has to decide whether (1) to hand the content over to another node, (2) to drop the copy, or (3) to replicate the content and hand over both copies. We refer to the nodes hosting a content copy at a given time instant as *replica nodes*, and we denote their set by $\mathcal{C}(t)$. Only replica nodes are responsible for updating the content and for injecting a new version in the wireless network.

Next, to highlight our contribution with respect to previous work, we relate our study to the formulation of the replication and replica placement problems typically used in the literature. Let us fix the time instant and drop the time dependency for ease of notation. Then, let $G = (V, E)$ represent the network graph at the given time, defined by a node set V and an edge set E. Let $\mathcal{C}$ denote the set of facility nodes, i.e., nodes holding a content replica. The specification of the placement of a given number of replicas, k, amounts to solving the uncapacitated k-median problem, which is defined as follows.

DEFINITION 1. *Uncapacitated k-median. Given the node set V with pair-wise distance function d, service demand $\lambda_{v(j)}, \forall v(j) \in V$, select up to k nodes to act as facilities so as to minimize the joint cost $C(V, \lambda, k)$:*

$$C(V, \lambda, k) = \sum_{\forall v(j) \in V} \lambda_{v(j)} d(v(j), m(v(j)))$$

where $m(v(j)) \in \mathcal{C}$ is the facility that is closer to $v(j)$.

The replica node set $\mathcal{C}$, instead, can be obtained by solving the following uncapacitated facility location problem at a given time instant.

DEFINITION 2. *Uncapacitated facility location. Given the node set V with pair-wise distance function d, service demand $\lambda_{v(j)}$ and cost for opening a facility at $v(j)$ $f(v(j))$, $\forall v(j) \in V$, select a set of nodes to act as facilities so as to minimize the joint cost $C(V, \lambda, f)$ of acquiring the facilities and servicing the demand:*

$$C(V, \lambda, f) = \sum_{\forall v(j) \in \mathcal{C}} f(v(j)) + \sum_{\forall v(j) \in V} \lambda_{v(j)} d(v(j), m(v(j)))$$

where $m(v(j)) \in \mathcal{C}$ is the facility that is closer to $v(j)$.

For general graphs, both the above problems are NP-hard [10] and a variety of approximation algorithms have been developed, which however require global (or extended) knowledge of the network state [9].

Which new problems are introduced in the context of our work? (i) Node mobility introduces the problem of a dynamic graph G, requiring that the facility location problem be solved upon every network topology or demand rate change. (ii) Even under static topology and constant demand, solving the facility location problem does not yield load balancing among nodes. (iii) The input to the facility location problem is the content demand workload generated by users: both replicas location and the number of replicas to deploy in a network depend on content consumption patterns. While the approach traditionally adopted is to assume content demand to be directed to the closest facility, as stated in Defs. 1 and 2, the wireless nature of our system allows content requests to propagate in the network, potentially reaching multiple facilities (replica nodes).

Our main contribution is therefore the design of a mechanism for content placement and replication that achieves load balancing as the network topology and the query rate vary, while taking into account the implications of query propagation towards replica nodes.

3. DISTRIBUTED MECHANISM FOR REPLICATION AND PLACEMENT PROBLEMS

We now outline our content distribution and replication procedures. Firstly, several techniques for query distribution and content access are detailed; next, we examine the challenging problem of replica placement, i.e., of which nodes are to be selected as carriers of content replicas to achieve load balancing; finally, we discuss the behavior of replica nodes as a function of the system workload, in search of a cooperative, distributed content replication strategy in presence of changing demand.

3.1 Content access mechanisms

The workload experienced by a replica node is determined by the mechanism used by nodes to access the content through device-to-device communications. We identify two phases: a content query transmission, and a query reply transmission (by the replica node carrying the desired content). We investigate several mechanisms for content access focusing on the content query transmission phase, and we assume that the identity of the nodes that have relayed the query is added to the query message itself. After a replica node with the desired content is found, it will reply to the node issuing the query through a multihop transmission process that backtracks the path from the replica node to the querying node, exploiting the identity of relay nodes included in the query message. This backtracking, although possibly occurring through multiple hops, makes no use of ad hoc routing protocols, as it is completely application-driven.

As far as the query transmission phase is concerned, the following three mechanisms are envisioned.

Scoped-flooding: content requests are simply flooded with a limited scope using application-layer broadcast. The "scope" can be defined as the maximum number of hops through which a query propagates, i.e., neighboring nodes propagate a query until it has traversed a maximum number of hops H, after which it is dropped. Clearly, if the request is received by a replica node, the content is served and the query is not propagated any further.

The main drawback of flooding is that multiple content replicas within reach of a node will be "hit" by a request. Beside causing congestion when a large number of replica nodes reply to the querying node, this also creates an artificially inflated workload, which conflicts with the underlying assumptions in Defs. 1 and 2. In our experiments, we explore the benefits of a *selective reply* mechanism that replica nodes can use to mitigate excessive workloads due to flooding. When selective reply is enabled, a replica node replies to a query with a probability that is inversely proportional to the hop-count of query messages.

Scanning: instead of flooding in all directions, the node issuing the query specifies an angular section within which the query is to be propagated by other nodes. In order to do so, it includes its own position (e.g., obtained through GPS), and the angle boundaries. All nodes receiving the query rebroadcast it only if their position satisfies the angular requirements, until a replica node is found or the query has traversed a maximum number of hops H. Nodes that are not within the angular section specified in the query will discard the message. If no reply is received after a timeout, a new sector is scanned, and the scanning of all sectors is repeated till either a reply is received or a maximum number of retries has been achieved. The number of sectors S, each of width $2\pi/S$, is a parameter of the system.

The complexity of this mechanism is comparable to that of scoped-flooding, however we will show that it reduces the overhead experienced with flooding. On the downside, scanning requires nodes to be able to estimate their position and reduces the probability of solving a query with respect to flooding-based solutions. Indeed, when a replica is within the sector currently scanned by the requesting node but it is farther than one hop away, one or more relay nodes would be needed to reach the replica. However, if at least one of the available relays are located outside the sector, the replica is not reached and the content query remains unsolved. Thus, the narrower the sector, the more likely that the query is unsuccessful.

Perfect-discovery: in this case, which is added for comparison purposes, nodes are assumed to be able to access a centralized content-location service that returns the identity of the closest content replica in terms of euclidean distance. We do not address the problem of how the centralized service is updated, save by noting that it is certainly responsible for additional overhead and complexity, and that it can be

managed through a separate protocol using unicast or multicast transmissions. A query is propagated using application-driven broadcast, but only the intended replica node (specified in the query) will serve the content. Any other replica node will discard the request.

On the one hand, this content access mechanism is the most demanding because it requires the presence of an auxiliary service to discover the closest replica. On the other, only one replica node carries the workload generated by the closest users, which is the hypothesis to the optimization problems stated in Sec. 2.

Finally, we improve the query/reply propagation process by adopting the PGB technique [11] for selecting forwarding nodes and sequence numbers to detect and discard duplicate queries.

3.2 Replica placement

Next, we overview the distributed lightweight algorithm that we use to solve the replica placement problem. Recall that any mobile device can be selected to host a content replica for a limited amount of time, that we term *storage time*, τ.

Also, as the first step to our study, we focus on the case of homogeneous user query rate, i.e., $\lambda_{v(j)}(t) = \lambda(t), \forall v(j)$.

As discussed in Sec. 2, at a fixed time instant, replica placement can be cast as the uncapacitated k-median problem. Given a set of potential locations to place a replica, the problem is to position an a-priori known number k of replicas according to Def. 1, i.e., so as to minimize the distance between replica node and requesting node. For a generic distribution of nodes over the network area, the solution of the k-median problem for different instances of the network graph yields replica placements that are instances of a random variable uniformly distributed over the graph. This is quite an intuitive result, confirmation of which we found by applying the approximation algorithm in [9] to the solution of the k-median problem in presence of various network deployments.

As pointed out earlier, though, the solution in [9] cannot be applied to our case since it is centralized and requires global knowledge of the network. We therefore devise a lightweight distributed mechanism that well approximates a uniform distribution of the replicas over the network nodes, i.e., a *nodal uniformity*, and that allows users to take turns in playing the role of replica nodes so as to achieve load balancing.

According to our mechanism, named *Random-Walk Diffusion (RWD)*, at the end of its storage time, a replica node selects with equal probability one of its neighbors to store the content for the following storage period. Thus, content replicas roam the network by moving from one node to another, randomly, at each time step τ.

To understand the extent to which replica placement achieved by our simple technique resembles the target nodal distributions, in Sec. 4 we employ the well-known χ^2 goodness-of-fit test on the inter-distance between content replicas. Whenever the computation complexity allows us, we compare the temporal evolution of the inter-distance distribution of replicas obtained by our scheme against the optimal replica placement computed by solving the k-median problem. Otherwise, we consider as term of comparison the empirical distribution of the distance between two nodes measured in simulation. Note that using inter-distances instead of actual coordinates allows us to handle a much larger number of samples (e.g., $|V| \cdot (|V|-1)$ instead of just $|V|$ samples) thus making the computation of the χ^2 index more accurate.

It is clear that the quality of approximation of the target replica distributions achieved by our store-and-forward mechanism depends on the node density: the higher the density, the better our approximation.

3.3 Content replication

We now focus on the more general problem of the uncapacitated facility location, defined in Sec. 2, where the optimal number of replicas (facilities) to be placed in the network is to be determined along with their location. In particular, we want to answer the following questions.

1. Given a set of demand points that exhibit a homogeneous query rate λ, what is the optimal number of content replicas that should be deployed in the network to achieve load balancing?

2. Is it possible to design a lightweight distributed algorithm that approximates this optimal number of replicas in presence of a dynamic demand and time-varying topology?

We address these questions by suggesting simple modifications to the RWD mechanism described in Sec. 3.2.

Again, we fix the time instant and, for simplicity, we drop the time dependency from our notation. Let the network be described by the graph $G = (V, E)$, with $|V| = N$ nodes deployed on an area $\mathcal{A}$. Also, recall that $\mathcal{C}$ and $V \backslash \mathcal{C}$ represent the sets of content replicas and of nodes issuing requests, respectively.

Given G and the query rate λ, the uncapacitated facility location problem amounts to the joint optimization of the number of replicas and their locations in the network. The RWD mechanism achieves a good approximation of the optimal placement in mobile networks, but ignores the cost to deploy a content replica. Now, with reference to Def. 2, we define the cost function to deploy content replicas in the network $f(v(j)), \forall v(j) \in \mathcal{C}$, as follows:

$$f(v(j)) = |s_{v(j)} - s_R| \tag{1}$$

where $s_{v(j)}$ is the workload expressed as number of queries served by replica node $v(j)$ during its storage time, and s_R is a reference value for the workload that node $v(j)$ is willing to support. We assume the case where all replica nodes are

willing to serve the same amount of queries, although our study can be easily extended to the case of different values of s_R. Eq. (1) indicates that the cost for replica node $v(j)$ grows with the gap between its workload and the reference value s_R. By using the cost function in (1) in the facility location problem in Def. 2, we can determine the location and number of replicas so that load balancing is achieved under the idealistic assumption that each query reaches one replica only.

Our replication mechanism only involves replica nodes, which are responsible to decide whether to replicate, hand over or drop content based on local measurements of their workload. During storage time τ, the generic replica node $v(j)$ *measures* the number of queries that it serves, i.e., $\hat{s}_{v(j)}$. When the storage time expires, the replica node compares $\hat{s}_{v(j)}$ to s_R. Decisions are taken as follows:

$$\text{if } \hat{s}_{v(j)} - s_R \begin{cases} > \epsilon & \text{replicate} \\ < -\epsilon & \text{drop} \\ \text{else} & \text{hand over} \end{cases}$$

where ϵ is a tolerance value to avoid replication/drop decisions in case of small changes in the node workload.

The rationale of our mechanism is the following. If $\hat{s}_{v(j)} > s_R$, replica node $v(j)$ presumes the current number of content replicas in the area to be insufficient to guarantee the expected workload s_R, hence the node replicates the content and hands the copies over two of its neighbors (one each), following the RWD placement mechanism (Sec. 3.2). The two selected neighbors will act as replica nodes for the subsequent storage time. Instead, if $\hat{s}_{v(j)} < s_R$, replica node $v(j)$ thinks that the current number of replicas in the area is exceeding the total demand, and just drops the content copy. Finally, if the experienced workload is (about) the same as the reference value, $v(j)$ selects one of its neighbors to hand over the current copy.

We stress that replication and placement are tightly related. For example, if content demand varies in time or in space (e.g., only a fraction of all nodes located in a subzone of the network area issue queries), both the number of replicas and their location must change. Thanks to the fact that replica nodes take decisions based on the measured workload, our solution can dynamically adapt to a time- or space-varying query rate, as will be shown by our simulation results. On the contrary, when the content demand is constant and homogeneous, our handover mechanism ensures load balancing among the network nodes.

In the following, we set up a simulation environment to evaluate the behavior of our mechanism when the wireless network is both static and dynamic. We also characterize the time the system takes to reach an optimal number of content replicas and we investigate the impact of the content access scheme on the performance of our solution.

4. SIMULATION-BASED EVALUATION

We implemented our replica placement and content replication mechanism in the ns-2 simulator. For each experiment described in the following, we execute 10 simulation runs and report averaged results. Our statistics are collected after an initial warm-up period of 500 s.

In our simulations, which lasted for almost 3 hours of simulated time (10000 s), we assume nodes to be equipped with a standard 802.11 interface, with an 11 Mbps fixed data transmission rate and a radio transmission range of 20 m. We consider a single content, whose size is of the order of 1 KB. In our evaluation we do not simulate cellular access. We point out that all standard MAC-layer operations are simulated, which implies that both queries and replies may be lost due to typical problems encountered in 802.11-based ad hoc networks (e.g., collisions or hidden terminals). This explains why, in the following, even nominally "ideal" access techniques may not yield the expected good performance.

We focus our attention on wireless networks with high node density: we place $N = 320$ nodes uniformly at random on a square area $\mathcal{A}$ of 200×200 m^2, with a resulting average node degree of 9–10 neighbors. We simulate node mobility using the *stationary* random waypoint model [12] where the average node speed is set to 3 m/s and the pause time is set to 100 s. These settings are representative, for example, of people using their mobile devices as they walk.

Unless otherwise stated, the parameters that define the mechanisms described in this work are set as follows. For the content access mechanisms, we set the scope of flooding and scanning to $H = 5$ hops: e.g., a node can cover half of the network diameter with scoped-flooding. In the case of scoped-flooding or perfect-discovery, if a query fails (i.e., no answer is received after 2 s), a new request is issued, up to a total of 5 times. If the scanning mechanism is used, a complete scan of 2π is divided into $S = 5$ angular sectors, each of which is visited for a maximum of 0.5 s, at most 5 times[1].

Finally, the tolerance value ϵ used in the replication/drop algorithm is equal to 2, unless otherwise stated; for all nodes, the storage time τ is set to 100 s, the user request rate is $\lambda = 0.01$ req/s, and the reference workload for a replica node is equal to $s_R = 10$.

4.1 Results

We present the main results of our work organized in a series of questions. We focus on the mobile scenario, but present results for a static network when the comparison is relevant. Tab. 1 summarizes the notations used in our figures to refer to content access mechanisms.

How well does our replica placement approximate the target distribution?

Here we assume a known number of content replicas to be deployed ($|\mathcal{C}|$=30), i.e., we consider the k-median problem discussed in Sec. 2. We measure the accuracy of our dis-

[1]We use the parameters that give the best results in terms of content access performance.

Table 1: Notation for different content access mechanisms. The post-fix "S" and "M" indicate a static and a mobile network, respectively.

Content access mechanism	*Notation*
Perfect-discovery	PS, PM
Scanning	SS, SM
Scoped-flooding	FS, FM
Scoped-flooding with selective reply	FS*,FM*

tributed replica placement mechanism using the χ^2 goodness-of-fit test on the inter-distance between replicas, as explained in Sec. 3.2. In case of a mobile network, we compute the distribution of replica nodes as follows: every τ seconds we take a snapshot of the network in its current state, we compute a reference distribution (e.g., nodal uniformity) of content replicas and use the χ^2 test against the distribution achieved by our mechanism. We remark that lower values of the χ^2 index indicate a better approximation, and that usually a value of χ^2 below 3.84 is considered a good fit [13].

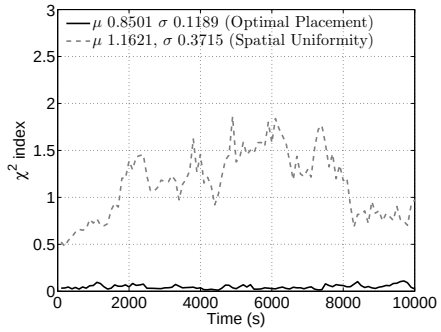


Figure 1: Temporal evolution of the χ^2 index in a static scenario ($|\mathcal{C}|$=30 and τ=100 s).

We first focus on a static network in which nodes are uniformly distributed on $\mathcal{A}$. Fig. 1 shows that our scheme does an excellent good job of approximating the optimal replica placement[2]. (We omit the comparison to nodal uniformity since our results show that the values of the χ^2 index are practically identical.) We therefore asked ourselves if a similar match could be found if the replica placement were uniformly distributed in space. As can be seen in the figure, a higher value of χ^2 indicates a poorer match with our placement scheme, mainly due to the lack of nodes (hence of replicas) where the spatially uniform distribution would have theoretically placed them.

Fig. 2 depicts the χ^2 test against *nodal uniformity* for the mobile scenario. Note that, due to the *stationary* random way-point mobility model used in our simulations, the node distribution is not uniform on the network area. The temporal evolution of the χ^2 index suggests that our replica placement mechanism is able to approximate very well nodal uniformity, despite network dynamics. We did not consider the optimal placement in this case, due to the cumbersome computational load.

[2]In fact, we solve the k-median problem through the centralized local-search algorithm described in [9] and obtain a tight approximation of the optimal solution.

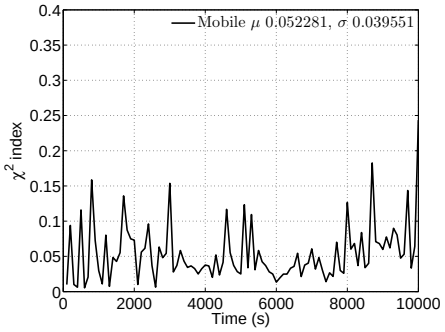


Figure 2: Temporal evolution of the χ^2 index in a mobile scenario ($|\mathcal{C}|$=30 and τ=100 s).

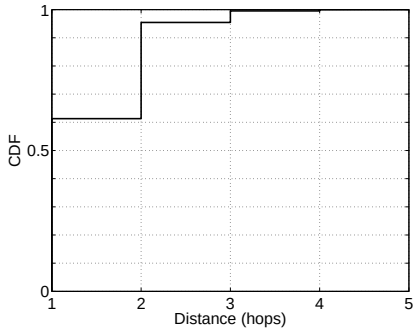


Figure 3: CDF of the hop-distance to the closest content replica in a mobile scenario ($|\mathcal{C}|$=30 and τ=100 s).

Finally, Fig. 3 reports the cumulative distribution function (CDF) of the distance (in number of hops) between a content consumer and the closest replica. We observe that more than 50% of nodes can reach the closest replica within 1 hop, and more than 90% of nodes are within 2 hops from the closest replica, i.e., for $|\mathcal{C}|$=30, consumers can access content within very few hops.

***Summary**: we overviewed our replica placement mechanism as a necessary introductory step to the replication scheme. We showed that the RWD mechanism can approximate very accurately the optimal solution to the k-median problem (Fig. 1), and it clearly approximates nodal uniformity even when the network is dynamic (Fig. 2). For the placement problem alone, in [8] we also explored the implications of clustered networks, different mobility models, and the parameter τ.*

What is the performance of content access mechanisms?

We evaluate the performance of the four content access mechanisms listed in Tab. 1, in terms of the following metrics:

- *solving ratio*, i.e., the ratio of satisfied requests to the total number of queries generated in the network. The target value is 1, corresponding to 100% of solved queries;
- *reply redundancy*, i.e., the number of replies to the same request, received from different replica nodes. The target value is 1, corresponding to one reply to each query;
- *latency*, i.e., the delay experienced by nodes to access information.

Fig. 4 shows the following quantiles of the access performance metrics for $|\mathcal{C}|$=30: the 25% (resp. 75%) as the lower (resp. higher) boundary in the error box, the 50% as the line within the error box. The brackets above and below the error box delimit the support of the CDF for that metric. For all access mechanisms, the median solving ratio (Fig. 4.a) is higher than 0.9, which indicates that only a small fraction of queries cannot reach a content replica. We observe that node mobility helps improving the solving ratio. The scheme that exhibits a slightly worse performance appears to be the scanning scheme, which is seldom unabke to reach a replica through relay nodes (see Sec. 3).

Fig. 4.b depicts the extent to which flooding-based mechanism can artificially inflate the workload of replica nodes: in our experiments, a single query can hit almost 6 replicas in the worst case[3]. High redundancy has a direct consequence on the behavior of the replication mechanism, as we discuss in detail later. We observe that the selective reply mechanism can halve the level of redundancy typical of flooding, and that node mobility helps in reducing redundancy in all schemes. It is important to notice that the scanning mechanism achieves a low reply redundancy, without requiring the presence of an auxiliary mechanism to help consumer nodes target the closest replica.

Latency for each content access scheme is shown in Fig 4.c: scanning is clearly the outlier in this figure, and we observe that node mobility might introduce additional delays.

We now provide more details on the performance of the scanning mechanism. Fig. 5 shows the impact of the time spent waiting for a reply on each sector composing the scanning horizon; we term this time *sector timeout*. The solving ratio is marginally affected by this parameter for a static network, but it decreases in the mobile case. Indeed, delaying the search in the next sector by a longer time has the mobile node skip larger portions of the area: two consecutively scanned sectors turn out to be non-adjacent due to the change in the position of the node issuing the query. The redundancy decreases with longer sector timeouts: indeed, the longer the timeout the higher the probability that a node scans another sector, i.e., it issues another query, only when no replica is available in the current sector. Instead, the latency deteriorates with a longer sector timeout because it will take more time to hit the sector where the replica is located. Mobility seems to have a positive effect on the delay, even with longer sector timeouts, since most solved queries are due to closeby replica nodes (farther nodes may reply after the querying node has moved away).

Fig. 6 shows the impact of the number of angular sectors in which the space around a node is partitioned, as determined by the scanning angle parameter. As explained in Sec. 3.1, a small scanning angle might reduce the probability for a query to reach a replica, hence the lower solving ratio with small angles. We observe a similar effect on redundancy: smaller angles limit the number of replicas "hit" by a query. Instead, the latency decreases with larger scanning angles because the probability to find a replica within a sector increases.

Summary: *we analyzed the performance of several content access mechanisms, ranging from simple flooding-based to complex schemes requiring perfect-discovery. With the setting used in our tests, we showed that a content query hits at least one replica with very high probability (Fig. 4) and that access delay can be slightly larger than 1 s with the scanning mechanism. Despite having larger delays, our results (Figs. 5 and 6) showed that the scanning mechanism achieves very low redundancy (comparable to perfect-discovery) and bears little costs in terms of complexity (which is comparable to flooding).*

Is the replication mechanism effective in reaching a target number of replicas?

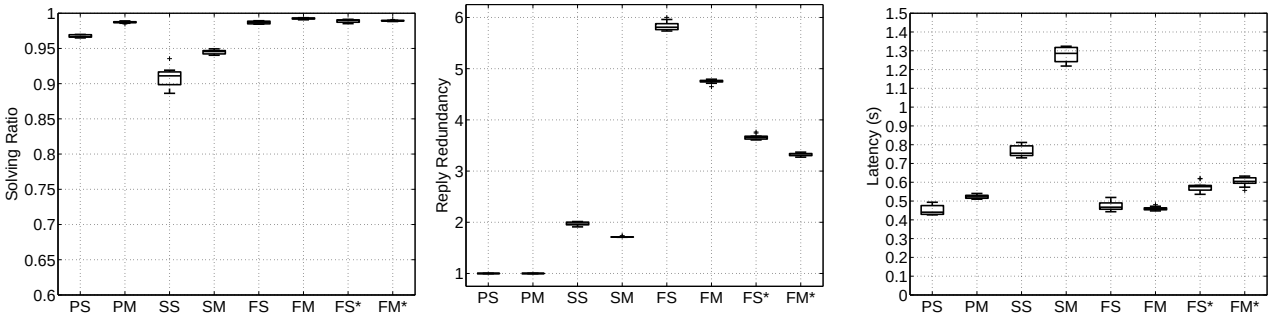
We now turn our attention to the *uncapacitated facility location* described in Sec. 2 and study how well the replication mechanism defined in Sec. 3.3 approximates the joint problem of replication and placement. Recall that the hypothesis of content demand points to be associated to the closest facility is hardly viable in the context of broadcast wireless networks (only the perfect-discovery scheme achieves it but with significant additional complexity).

Here we take an extreme scenario in which only one copy of the content is initially present in the network and we focus on the evolution in time of the number of replicas in the system. We omit the temporal evolution of the χ^2 index, since our results are consistent with what we have observed for the placement scheme without replication.

Fig. 7 shows the temporal evolution of the total number of content replicas $|\mathcal{C}|$ for the mobile scenario. In the plot, we report a reference line representing the target number of content replicas computed as follows. Let us consider the perfect-discovery content access mechanism: demand points are associated to their closest replica. Finding the optimal number of content replicas amounts to solving the uncapacitated facility location problem for a given network graph. We have implemented the *centralized* algorithm in [9][4], which is a rigorous but very demanding approach when

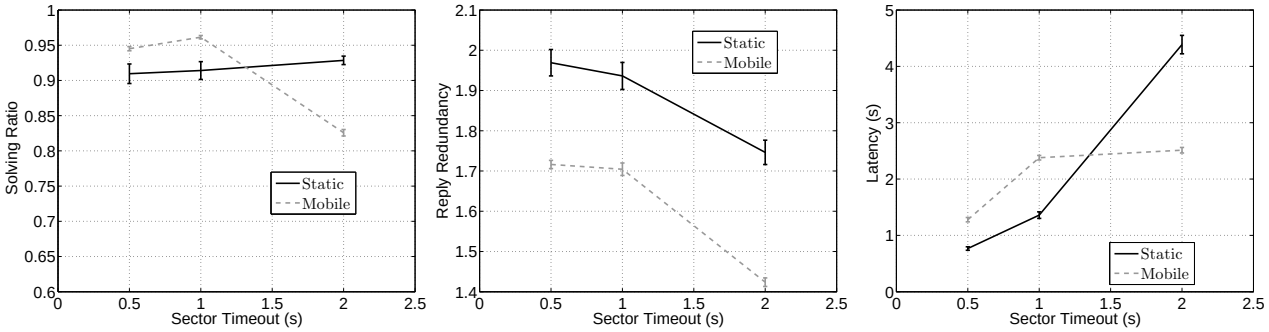
[3]We also run experiments with lower values of the flooding scope H: redundancy, which is proportional to H, remains higher in flooding compared to other schemes.

[4]We set a non-uniform cost to open a facility proportional to its



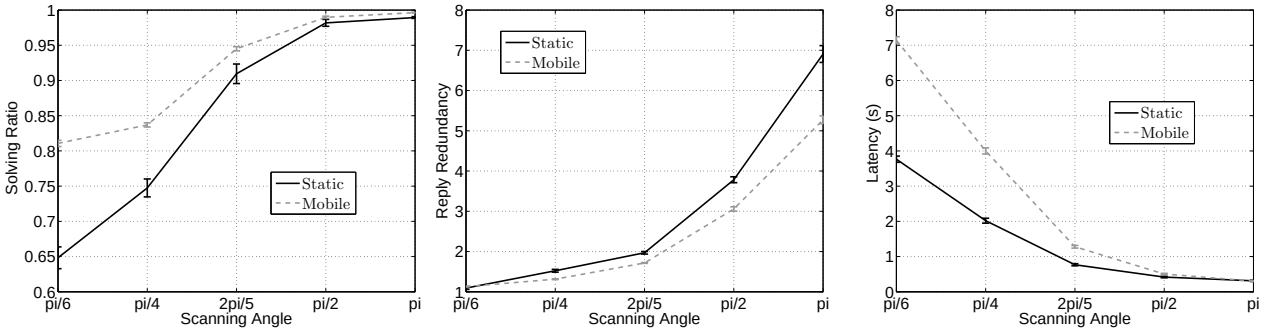
(a) Solving ratio (b) Reply redundancy (c) Latency

Figure 4: Performance of content access mechanisms, in a static and mobile scenario ($|\mathcal{C}|$=30 and τ=100 s).



(a) Solving ratio (b) Reply redundancy (c) Latency

Figure 5: Performance of the scanning mechanism as a function of the sector timeout (scanning angle $2\pi/5$, $|\mathcal{C}|$=30 and τ=100 s).



(a) Solving ratio (b) Reply redundancy (c) Latency

Figure 6: Performance of the scanning mechanism as a function of the scanning angle (sector timeout = 0.5s, $|\mathcal{C}|$=30 and τ=100 s).

the network is dynamic, despite a polynomial execution time. The optimal number of content replicas, however, can be approximated: assume each replica node to receive, on average, the same share of queries. Then, we would like the target number of replica nodes $|\mathcal{C}^*|$ to be such that:

$$|\mathcal{C}^*|\frac{s_R}{\tau} = (N - |\mathcal{C}^*|)\lambda \quad (2)$$

where $(N-|\mathcal{C}^*|)\lambda$ is the network aggregate query rate. From (2), we write:

$$|\mathcal{C}^*| = \frac{N\lambda\tau}{\lambda\tau + s_R} \quad (3)$$

For the parameters used in our simulations, the sample solution of the centralized algorithm and the value in (3) agree on the target number of replicas: the system should reach $|\mathcal{C}^*| = 30$ replica nodes.

Fig. 7 indicates that the number of content replicas we achieve with our scheme strikingly matches the target value when perfect-discovery is used: in steady state, the average relative error is less than 2%. We can also observe the adverse effects of the artificially inflated workload created by the scanning and flooding mechanisms. In case of flooding, the number of replicas in the system is drastically overestimated. Despite its simplicity, the scanning mechanism induces a much smaller error than flooding in reaching the target number of content replicas.

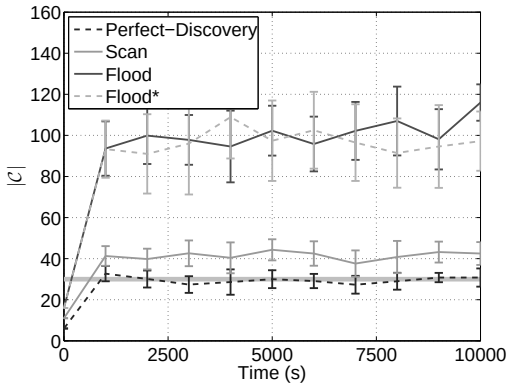


Figure 7: Temporal evolution of the number of replicas, for a network bootstrapping with $|\mathcal{C}| = 1$ in a mobile scenario ($\lambda = 0.01$, $s_R = 10$, $\tau = 100$ s, $|\mathcal{C}^*| = 30$).

Fig. 8 depicts the ratio between the aggregate number of replication and drop decisions[5] and shows in more details the behavior of the replication mechanism: when a single or few nodes support most of the content queries, the number of replication decisions is considerably higher than drop decisions. Once a sufficient number of replicas have populated the network, the ratio reaches the steady state value of 1.

degree: indeed, a highly connected node will most likely attract more demand from content consumers.

[5]For sake of clarity, in this figure we omit handover decisions and we report results every 1000 s.

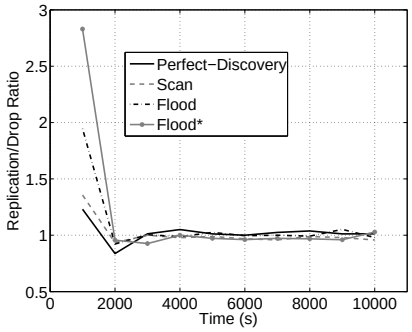


Figure 8: Replication/drop ratio for a mobile scenario for a network bootstrapping with $|\mathcal{C}| = 1$ ($\lambda = 0.01$, $s_R = 10$, $\tau = 100$ s).

How is the total workload shared among replica nodes?

As before, we study the joint placement and replication problem and we use the extreme scenario in which the network is initialized with only one content replica.

Fig. 9 shows the 25%, 50% and 75% quantiles of the workload for each replica node, aggregated over the simulation time. The figure is complemented with the *average* workload per replica node. The reference value $s_R = 10$ is shown as a horizontal line in the plot. As expected, with perfect-discovery the average load matches the reference value s_R, both in the static and mobile scenario. Instead, when flooding is used, the average load is consistently above s_R, albeit the "approximation error" is smaller than 3%. It should be noted, with reference to Fig. 7, that, when flooding is used, replica nodes achieve a good approximation of s_R because the total number of content replicas is higher than the target value: since the workload induced by flooding reaches multiple replica nodes, our adaptive scheme replicates more than necessary to maintain the workload within the desired target s_R. When scanning is used, the quality of approximation of s_R is excellent. Node mobility helps in reducing the skewness of the workload distribution. We note that it is possible that some nodes experience very little workload: this happens when the location of a replica is very close to the boundaries of the node deployment area, which implies that fewer content queries will reach those replica nodes.

What is the convergence time of the replication mechanism?

Convergence time should be carefully defined in our context: clearly, our mechanism cannot settle to a static, unique content replica placement, nor can it stabilize on a unique number thereof. For placement, it is not our intent to statically assign the role of content replica to a node and deplete nodal resources: we seek to balance the workload across all network nodes. We assume the network to have converged to a steady state when the difference between the reference

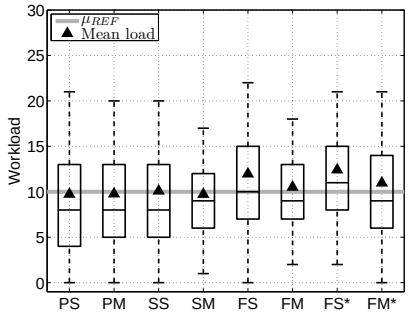


Figure 9: Aggregate workload distribution for replicas for a network bootstrapping with $|\mathcal{C}| = 1$ ($\lambda = 0.01$, $s_R = 10$, $\tau = 100$ s).

value computed through (3) and the experimental number of replicas is within 2%.

Again, we consider a *worst-case* scenario in which only one copy of the content is initially present in the network. Tab. 2 illustrates how convergence time (labelled t_s) varies with the storage time τ, and the tolerance value ϵ in the case of perfect-discovery. We also performed experiments to study the impact of the network size: we have observed a linear growth of the convergence time with N. Since the storage time τ is used to trigger replication/drop decisions, we expect to see a positive correlation between τ and convergence time: Tab. 2 confirms this intuition. We note that there is a trade-off between the convergence time, and the message overhead: a small storage time shortens the convergence time at the cost of an increased number of content movements from a node to another. Moreover, in our simulations we do not trace the message overhead required by the perfect discovery mechanism: with frequent reassignments of a node to the closest replica, this overhead could become prohibitive. As for the impact of the tolerance parameter ϵ, our experiments indicate that a very reactive scheme would yield smaller convergence times, at the risk of causing frequent oscillations around a target value.

Table 2: Average convergence time t_S as a function of the storage time τ ($\epsilon = 2$) and the tolerance factor ϵ ($\tau = 100$ s), with perfect-discovery.

τ (s)	t_S (s)
20	800
50	1500
100	1700
150	2000
200	2300

ϵ	t_S (s)
0	700
2	1700
5	1900

Summary: *we showed that our replication mechanism, which provides for content copies to by added, dropped or handed over by replica nodes achieves a very good approximation of the optimal number of replicas (Fig. 7) and a target placement thereof (Fig. 9), in a variety of scenarios and under several content access mechanisms (Fig. 8) . Our scheme is robust against mobility, which turns out to be an ally especially for balancing the workload among replica nodes. We also studied the parameters that influence the time it takes for the network to reach a steady state and discussed the tradeoff that exists between convergence time and message overhead (Tab. 2). The scanning mechanism suggested in this work mitigates the message overhead due to an auxiliary service to support perfect discovery, at the cost of increasing the estimation error in the number of content replicas to place in the network.*

What is the impact of variations in time and in space of the content demand?

We now focus our attention on the behavior of content replication in presence of a dynamic workload. We first examine workload variations in time. In a first phase, that begins at time 0 and ends at 5000 s, we set the content request rate as $\lambda = 0.01$ req/s. In a second phase, from 5000 s to the end of the simulation, the request rate doubles, i.e., $\lambda = 0.02$ req/s.

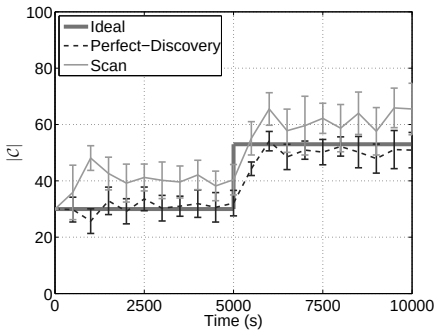


Figure 10: Temporal evolution of the number of replica nodes in case of variations in time of the content demand, for a mobile network. $|\mathcal{C}^*|$ is equal to 30 and 53 in the first and second phase, respectively.

Fig. 10 shows the temporal evolution of the number of replicas in a mobile network. The figure is enriched with two reference values[6]: in the first phase $|\mathcal{C}^*| = 30$, in the second phase $|\mathcal{C}^*| = 53$. We report results for the perfect-discovery and the scanning access schemes: our mechanism achieves a very good approximation of the target number of replicas with perfect-discover, and slightly over-replicates the content when scanning is used. Despite node mobility, not only is our scheme able to correctly determine the number of replicas but also their target location; as a consequence, the load distribution is minimally affected by a variation in time of content demand. This result is shown in Fig. 11, where we indicate the 25%, 50% and 75% quantiles of the workload, and we report the average load per replica node. Note that, although at first glance the mean load values

[6]As explained before, we compute the target values through (3), but compare them with the solution of the facility location problem computed over several snapshots of the network graph.

look identical, there is a minimal difference among them, as shown by the numbers above the error bars in the plot.

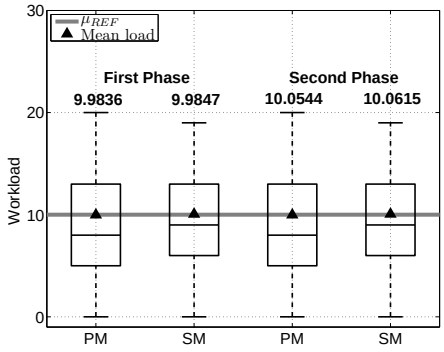


Figure 11: Workload distribution of replica nodes for variations in time of the content demand, in a mobile network.

We now turn our attention to variations in space of content demand: we describe the behavior of the content replication mechanism with the following example. For the initial 5000 s of the simulation time, content queries are issued by all nodes deployed on the network area $\mathcal{A}$ of size 200 m^2. Subsequently, we select a smaller square area α of size 100 m^2 and instruct only nodes within that zone to issue content queries, while all other nodes exhibit a lack of interest. Nodes use perfect-discovery to access the content.

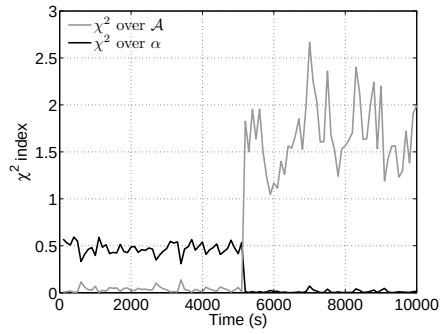


Figure 12: Temporal evolution of the χ^2 index for variation in space of the content demand, in a mobile network.

Fig. 12 shows the temporal evolution of the χ^2 index computed as follows. We compare the distribution of content replicas achieved by our mechanism against the target *nodal uniformity* distribution computed over the network area $\mathcal{A}$ and the sub-zone α, and plot the two curves. When all nodes issue queries, the line of the χ^2 index computed over $\mathcal{A}$ indicates a good approximation of the target replica placement. This is true up to roughly 5000 s, after which the χ^2 in $\mathcal{A}$ shows a substantial deterioration in approximating nodal uniformity on all nodes. Indeed, after 5000 s, the target replica placement should be computed over the area α, and the second line of the χ^2 index computed over α indicates that the distribution of replicas achieved by our mechanism is indeed a good approximation of a target placement over α. We can conclude that our mechanism allows content replicas to "migrate" where the demand is higher.

***Summary**: we showed that our mechanism achieves a target placement and a sufficient number of content replicas to cope with complex demand scenarios, even in a mobile network. When content demand varies in time, our mechanism adaptively replicates the content to meet the variation in the workload (Fig. 10). When content demand varies in space, our scheme allows content replicas to migrate to the location where the demand is higher and meet a variation in the workload (Fig. 12).*

5. RELATED WORK ON REPLICATION IN MULTIHOP NETWORKS

Simple, widely used techniques for replication are gossiping and epidemic dissemination [14, 15], where the information is forwarded to a randomly selected subset of neighbors. Although our RWD scheme may resemble this approach in that a replica node hands over the content to a randomly chosen neighbor, the mechanism we propose and the goals it achieves (i.e., approximation of nodal uniformity and optimal number of replicas) are significantly different.

Another viable approach to replication is represented by quorum-based [16] and cluster-based protocols [17]. Both methods, although different, are based on the maintenance of quorum systems or clusters, which in mobile network are likely to cause an exceedingly high overhead. Node grouping is also exploited in [18, 19], where groups of nodes with stable links are used to cooperatively store contents and share information. The schemes in [18, 19], however, require an a-priori knowledge of the query rate, which is assumed to be constant in time. Note that, on the contrary, our lightweight solution can cope with a dynamic demand, whose estimate by the replica nodes is used to trigger replication. We point out that achieving content diversity is the goal of [20] too, where, however, cooperation is exploited among one-hop neighboring nodes only.

Threshold-based mechanisms for content replication are proposed in [21, 22]. In particular, in [21] it is the original server that decides whether to replicate content or not, and where. In [22], nodes have limited storage capabilities: if a node does not have enough free memory, it will replace a previously received content with a new one, only if it is going to access that piece of information more frequently than its neighbors up to h-hops. Our scheme significantly differs from these works, since it is a totally distributed and extremely lightweight mechanism, which accounts for the content demand by other nodes and ensures a replica density that autonomously adapts to the changes in the query rate over time and space.

Finally, relevant to our study are the numerous schemes

proposed for handling query/reply messages; examples are [23], which resembles the perfect-discovery mechanism, and [24,25] where queries are propagated along trajectories so as to meet the requested information. Also, we point out that the RWD scheme was first proposed in our work [8]. That paper, however, besides being a preliminary study, focused on mechanisms for content handover only: no replication or content access were addressed.

6. CONCLUSIONS

We focused on content replication in mobile networks and we addressed the joint optimization problem of (i) establishing the number of content replicas to deploy in the network, (ii) finding their most suitable location, and (iii) letting users efficiently access content through device-to-device communications.

To achieve these goals, we proposed a distributed mechanism that lets content replicas move in the network according to random patterns: network nodes temporarily store content, which is handed over to randomly selected neighbors. Hence the burden of storing and providing content is evenly shared among nodes and load balancing is achieved. In our mechanism, replica nodes are also responsible for creating content copies or drop them, with the goal of obtaining an ideal number of content replicas in the network. The workload experienced by a replica node is the only measured signal we use to trigger replication and drop decisions.

We studied the above problems through the lenses of facility location theory and showed that our lightweight scheme can approximate with high accuracy the solution obtained through centralized algorithms. Clearly, network dynamics exact a high toll in terms of complexity to reach an optimal replication and placement of content, and we showed that our distributed mechanism can readily cope with such a scenario. Moreover, we removed the typical assumption of assigning content demand points to their closest replica and investigated several content access schemes, their performance, and their impact on content replication.

Lastly, we studied the flexibility of our scheme when content demand varies in time and in space: our experiments underlined the ability of our approach to adapt to such variations while maintaining accuracy in approximating an optimal solution.

Our next step will be to relax the assumption of a cooperative setting and analyze selfish replication with tools akin to game theory. In [26] we show that the system we study can be modeled as an anti-coordination game, and our goal is to understand how to modify or extend the ideas presented in this work to achieve strategy-proofness.

7. REFERENCES

[1] A. Derhab and N. Badache, "Data Replication Protocols for Mobile Ad-hoc Networks: A Survey and Taxonomy," *IEEE Communications Surveys & Tutorials*, vol. 11, no. 2, pp. 33–51, 2009.

[2] L. Yin and G. Cao, "Supporting Cooperative Caching in Ad Hoc Networks," *IEEE Trans. on Mobile Computing*, vol. 5, no. 1, pp. 77–89, Jan. 2006.

[3] B. Tang, H. Gupta, and S. Das, "Benefit-based Data Caching in Ad Hoc Networks," *IEEE Trans. on Mobile Computing*, vol. 7, no. 3, pp. 289–304, Mar. 2008.

[4] M. Fiore, F. Mininni, C. Casetti, and C.-F. Chiasserini, "To Cache or Not To Cache?," *IEEE Infocom*, Rio de Janeiro, Brazil, Apr. 2009.

[5] N. Laoutaris, G. Smaragdakis, K. Oikonomou, I. Stavrakakis, and A. Bestavros, "Distributed Placement of Service Facilities in Large-Scale Networks," *IEEE Infocom*, 2007.

[6] P. Nuggehalli, V. Srinivasan, C.-F. Chiasserini, and R.R. Rao, "Efficient Cache Placement in Multi-hop Wireless Networks," *IEEE/ACM Trans. on Networking*, vol. 14, no. 5, Oct. 2006.

[7] P. Mirchandani and R. Francis, "Discrete Location Theory," *John Wiley and Sons*, 1990.

[8] C. Casetti, C.-F. Chiasserini, M. Fiore, C.-A. La, and P. Michiardi, "P2P Cache-and-Forward Mechanisms for Mobile Ad Hoc Networks," *IEEE ISCC*, Sousse, Tunisia, July 2009.

[9] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit, "Local Search Heuristic for k-median and Facility Location Problems," *ACM STOC*, 2001.

[10] O. Kariv and S. Hakimi, "An Algorithmic Approach to Network Location Problems, Part II: p-medians," *SIAM Journal on Applied Mathematics*, vol. 37, pp. 539–560, 1979.

[11] V. Naumov, R. Baumann, and T. Gross, "An Evaluation of Inter-vehicle Ad Hoc Networks Based on Realistic Vehicular Traces," *ACM MobiHoc*, Florence, Italy, May 2006.

[12] J.-Y. Le Boudec, M. Vojnovic, "Perfect Simulation and Stationarity of a Class of Mobility Models," *IEEE Infocom*, Mar. 2005.

[13] *NIST/SEMATECH e-Handbook of Statistical Methods*/, `http://www.itl.nist.gov/div898/handbook/`, 2009.

[14] H. Hayashi, T. Hara, and S. Nishio, "On Updated Data Dissemination Exploiting an Epidemic Model in Ad Hoc Networks," *BioADIT*, 2006

[15] M. Hauspie, A. Panier, and D. Simplot-Ryl, "Localized Probabilistic and Dominating Set Based Algorithm for Efficient Information Dissemination in Ad Hoc Networks," *IEEE MASS*, Oct. 2004.

[16] J. Luo, J.-P. Hubaux, and P. T. Eugster, "PAN: Providing Reliable Storage in Mobile Ad Hoc Networks with Probabilistic Quorum Systems," *ACM MobiHoc*, 2003.

[17] H. Yu, H. Hassanein, and P. Martin, "Cluster-based Replication for Large-scale Mobile Ad-hoc Networks," *International Conf. on Wireless Networks, Communications and Mobile Computing*, June 2005.

[18] T. Hara, "Effective Replica Allocation in Ad Hoc Networks for Improving Data Accessibility," *IEEE Infocom*, Apr. 2001.

[19] T. Hara, Y.-H. Loh, and S. Nishio, "Data Replication Methods Based on the Stability of Radio Links in Ad Hoc Networks," *DEXA*, 2003.

[20] L. Yin and G. Cao, "Balancing the Tradeoffs between Data Accessibility and Query Delay in Ad Hoc Networks," *IEEE SRDS*, 2004.

[21] V. Thanedar, K. C. Almeroth, and E. M. Belding-Royer, "A Lightweight Content Replication Scheme for Mobile Ad Hoc Environments," *Network*, 2004.

[22] M. Shinohara, H. Hayashi, T. Hara, and S. Nishio, "Replica Allocation Considering Power Consumption in Mobile Ad Hoc Networks," *IEEE PERCOMW*, 2006.

[23] K. Chen and K. Nahrstedt, "An integrated Data Lookup and Replication Scheme in Mobile Ad Hoc Networks," *SPIE ITCom*, Aug. 2001.

[24] D. Braginsky and D. Estrin, "Rumor Routing Algorithm For Sensor Networks," *ACM WSNA*, 2002.

[25] J. B. Tchakarov and N. H. Vaidya, "Efficient Content Location in Wireless Ad Hoc Networks,"*Mobile Data Management*, 2004.

[26] P. Michiardi, C.-F. Chiasserini, C. Casetti, C.-A. La, and M. Fiore, "On a Selfish Caching Game," *ACM PODC* (Brief Announcement), Calgary, Canada, Aug. 2009